AMENDMENTS TO THE CLAIMS

Please make the following amendments to the claims:

1. (Original) A method for providing encryption for the rerouting of multi-media data flow packets, comprising the steps of:

assigning a sequence number to a first multi-media data flow packet received by a first endpoint, wherein said first multi-media data flow packet is within a series of multi-media data flow packets;

pseudo-randomly shuffling said sequence number of said first multi-media data flow packet; and

transmitting said pseudo-randomly shuffled sequence number to a second endpoint.

2. (Original) The method of claim 1, wherein said multi-media data flow packets are real-time multi-media data flow packets.

3. (Original) The method of claim 1, wherein said pseudo-random shuffling is performed via use of randomization code that is algorithmically predictable if a key to said randomization code is known.

4. (Original) The method of claim 1, wherein said series of multi-media data flow packets, including said first multi-media data flow packet, are assigned sequence numbers that are each pseudo-randomly shuffled prior to said transmitting step.

5. (Original) The method of claim 1, further comprising the step of pseudo-randomly shuffling a destination address of said first multi-media data flow packet.

6. (Original) The method of claim 5, wherein said destination address is a destination port address of said second endpoint.

7. (Original) The method of claim 4, further comprising the step of re-sequencing said series of multi-media data flow packets so that said re-sequenced multi-media data flow packets are transmitted from said first endpoint to said second endpoint in a random order.

8. (Original) The method of claim 7, wherein said re-sequenced multi-media data flow packets are transmitted within a predefined jitter buffer size.

9. (Original) The method of claim 1, further comprising the step of performing bit manipulation within said first multi-media data flow packet.

10. (Original) The method of claim 9, wherein said step of performing bit manipulation is performed by using a bitsize operation that is restorable.

11. (Original) The method of claim 10, wherein said bitsize operation uses a negation operator, such that every 1 bit becomes a 0 bit and every 0 bit becomes a 1 bit.

12. (Original) A system for providing encryption for the rerouting of multi-media data flow packets, comprising:

means for assigning a sequence number to a first multi-media data flow packet received by a first endpoint, wherein said first multi-media data flow packet is within a series of multi-media data flow packets;

means for pseudo-randomly shuffling said sequence number of said first multi-media data flow packet; and

means for transmitting said pseudo-randomly shuffled sequence number to a second endpoint.

13. (Original) The system of claim 12, wherein said multi-media data flow packets are real-time multi-media data flow packets.

14. (Original) The system of claim 12, wherein said means for pseudo-random shuffling performs said shuffling via use of randomization code that is algorithmically predictable if a key to said randomization code is known.

15. (Original) The system of claim 12, further comprising means for pseudo-randomly shuffling a destination address of said first multi-media data flow packet.

16. (Original) The system of claim 15, wherein said destination address is a destination port address of said second endpoint.

17. (Original) The system of claim 12, further comprising means for re-sequencing said series of multi-media data flow packets so that said re-sequenced multi-media data flow packets are transmitted from said first endpoint to said second endpoint in a random order.

18. (Original) The system of claim 17, wherein said re-sequenced multi-media data flow packets are transmitted within a predefined jitter buffer size.

19. (Original) The system of claim 12, further comprising means for performing bit manipulation within said first multi-media data flow packet.

20. (Original) The system of claim 19, wherein said means for performing bit manipulation uses a bitsize operation that is restorable.

21. (Original) The system of claim 20, wherein said bitsize operation uses a negation operator, such that every 1 bit becomes a 0 bit and every 0 bit becomes a 1 bit.

22. (Currently Amended) A system for providing encryption for the rerouting of multi-media data flow packets, comprising:

a first endpoint, connected to a second endpoint, wherein said first endpoint comprises;

a transceiver;

software stored within said first endpoint defining functions to be performed by said first endpoint; and

a processor configured by said software to perform the steps of [[,]] :

assigning a sequence number to a first multi-media data flow packet received by a first endpoint, wherein said first multi-media data flow packet is within a series of multi-media data flow packets;

pseudo-randomly shuffling said sequence number of said first multi-media data flow packet; and

transmitting said pseudo-randomly shuffled sequence number to a second endpoint.

23. (Original) The system of claim 22, wherein said multi-media data flow packets are real-time multi-media data flow packets.

24. (Original) The system of claim 22, wherein said multi-media data flow packets are real-time multi-media data flow packets.

25. (Original) The system of claim 22, wherein said pseudo-random shuffling is performed via use of randomization code that is algorithmically predictable if a key to said randomization code is known.

26. (Original) The system of claim 22, wherein said series of multi-media data flow packets, including said first multi-media data flow packet, are assigned sequence numbers that are each pseudo-randomly shuffled prior to said transmitting step.

27. (Original) The system of claim 22, wherein said processor is further configured by said software to perform the step of pseudo-randomly shuffling a destination address of said first multi-media data flow packet.

28. (Original) The system of claim 27, wherein said destination address is a destination port address of said second endpoint.

29. (Original) The system of claim 26, wherein said processor is further configured by said software to perform the step of re-sequencing said series of multi-media data flow packets so that said re-sequenced multi-media data flow packets are transmitted from said first endpoint to said second endpoint in a random order.

30. (Original) The system of claim 29, wherein said re-sequenced multi-media data flow packets are transmitted within a predefined jitter buffer size.

31. (Original) The system of claim 22, wherein said processor is further configured by said software to perform the step of performing bit manipulation within said first multi-media data flow packet.

32. (Original) The system of claim 31, wherein said step of performing bit manipulation is performed by using a bitsize operation that is restorable.

33. (Original) The system of claim 32, wherein said bitsize operation uses a negation operator, such that every 1 bit becomes a 0 bit and every 0 bit becomes a 1 bit.

34. (Original) A system for providing encryption for the routing of multi-media data flow packets, comprising:

a first endpoint connected to a second endpoint, wherein said second endpoint comprises:

a transceiver;

software stored within said second endpoint defining functions to be performed by said second endpoint; and

a processor configured by said software to perform the steps of:

unshuffling a pseudo-randomly shuffled sequence number received from said first endpoint, via use of an alogrithmic key; and

deriving a first data flow packet from said unshuffled sequence number, wherein said first data flow packet is within a series of data flow packets.

35. (Original) A system for providing encryption for the routing of data flow packets, comprising:

a first endpoint connected to a second endpoint, wherein said first endpoint comprises:

a transceiver; and

a controller programmed to perform the steps of:

assigning a sequence number to a first multi-media data flow packet received by a first endpoint, wherein said first multi-media data flow packet is within a series of multi-media data flow packets;

pseudo-randomly shuffling said sequence number of said first data flow packet; and

transmitting said pseudo-randomly shuffled sequence number to a second endpoint.

36. (Original) The system of claim 35, wherein said multi-media data flow packets are real-time multi-media data flow packets.

37. (Original) The system of claim 35, wherein said series of multi-media data flow packets, including said first multi-media data flow packet, are assigned sequence numbers that are each pseudo-randomly shuffled prior to said transmitting step.

38. (Original) The system of claim 35, wherein said controller is further programmed to perform the step of pseudo-randomly shuffling a destination address of said first multi-media data flow packet.

39. (Original) The system of claim 38, wherein said destination address is a destination port address of said second endpoint.

40. (Original) The system of claim 37, wherein said processor is further configured by said software to perform the step of re-sequencing said series of multi-media data flow packets so that said re-sequenced multi-media data flow packets are transmitted from said first endpoint to said second endpoint in a random order.

41. (Original) The system of claim 40, wherein said re-sequenced multi-media data flow packets are transmitted within a predefined jitter buffer size.

42. (Original) The system of claim 35, wherein said controller is further configured to perform the step of performing bit manipulation within said first multi-media data flow packet.

43. (Original) The system of claim 42, wherein said step of performing bit manipulation is performed by using a bitsize operation that is restorable.

44. (Original) The system of claim 43, wherein said bitsize operation uses a negation operator, such that every 1 bit becomes a 0 bit and every 0 bit becomes a 1 bit.